Using VRML In Construction Industry Applications

Robert Lipman* Kent Reed

Building and Fire Research Laboratory
National Institute of Standards and Technology

Abstract

This paper describes initial research using the Virtual Reality Modeling Language (VRML97) in construction industry applications. The modeling of steel structures and construction equipment as objects for inclusion in construction-site world models was studied. The ultimate goal is to provide three-dimensional web-based technologies for managing, accessing, and viewing construction project information.

CR Categories and Subject Descriptors: I.3.6 [Computer Graphics]: Methodology and Techniques: Interaction techniques, standards; I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism: Virtual reality; J.6 [Computer-Aided Engineering]: Computer-aided design (CAD)

Additional Keywords and Phrases: VRML, virtual environments, computer-integrated construction, steel structures, construction equipment.

1 INTRODUCTION

A principal characteristic of construction industry projects is the distributed nature of the project team. Effective project delivery depends on making current and correct information available to all the project participants, wherever they are and whenever they need it. Traditionally, an enormous amount of information has been communicated by means of paper documents, including 2D drawings with annotations and associated schedules. Recently, the industry has embraced many kinds of web technologies, but information access continues to be based on the paper document metaphor. True model-based information access can be achieved only in special cases where all the participants use proprietary systems that understand a common protocol.

The purpose of the research described in this paper was to determine the applicability of the Virtual Reality Modeling Language (VRML97) in construction industry applications. In principle, VRML is an open standard that offers the possibility of accessing many types of construction project information using readily available and well-accepted graphical user interfaces based on 3D visualization of a model rather than paper.

Many of the commercial CAD systems used by the construction industry are primarily geometry modelers, rather than object modelers. Regardless of the file format used to export a model (including VRML), they frequently export the 3D model as only a collection of surfaces representing the geometry that contains far too many polygons and unnecessary details. They also fail to preserve the aggregation of geometry elements into objects and the relationship between objects. There is no possibility of accessing and viewing information in the 3D model other than the geometry.

Intelligently constructed VRML representations of steel structures can be done in an object-like fashion. This paper describes a VRML Prototype used to create a Beam object that provides a simple way to specify thousands of different types of beams with only a beam designation and position; at the same time providing access to other non-graphical information sources. The prototype encapsulates the details of the implementation of the Beam object. Object-like VRML representations make it easier to update models or to extend the implementation of the object without having to change the model.

Developing useful construction equipment and materials and their behaviors, that can interact with each other and their environment is also described.

2 STEEL STRUCTURES

2.1 Characterization

Steel structures are characterized by a variety of basic structural elements including beams, columns, plates, and pipes. With the exception of plates, basic structural elements are typically much longer than their overall cross section dimensions. Complex structures, such as trusses, are built from a combination of basic structural elements. Mechanical and chemical process equipment are often built from a combination of basic structural elements as well.

A standard steel beam or column usually has a cross section that is either rectangular (tubing), circular (piping or solid bars), I-shaped (I-beams), C-shaped (channels), T-shaped (tees), or L-shaped (angles). An I-beam, whose cross section is shaped like the letter ‘‘I’’, has a web and two flanges. The web is the vertical part of the ‘‘I’’ and the flanges are the horizontal parts of the ‘‘I’’. The cross section dimensions of an I-beam are specified by the width and thickness of the flanges and the depth and thickness of the web. The cross section is specified in a similar manner for other element types. Compound cross sections are built from two structural elements. For example, compound beam can be constructed from an I-beam and a channel where the web of the channel is joined to the top flange of the I-beam.

The American Institute of Steel Construction (AISC) [1] specifies over 1700 standard cross sections for the typical steel elements listed above. Each structural member lists cross section dimensions and other properties relevant to a structural engineer. These properties include moments of inertia, the torsional con-

DISCLAIMER: Mention of trade names does not imply endorsement by NIST.

*National Institute of Standards and Technology
100 Bureau Drive, Stop 8630
Gaithersburg MD 20899-8630
email: robert.lipman@nist.gov, kent.reed@nist.gov
http://cic.nist.gov/vrml/

stant, and the nominal weight per foot. Each cross section has an alphanumeric designation. For example, a W24X146 section is an I-beam (historically, ''W'' is used for I-beams) that is approximately 61 cm ± 2.5 cm (24 in ± 1 in) deep and weighs about 217.3 kg/m (146 lb/ft).

Steel structural elements also have a variety of end conditions. Diagonal truss members have their ends cut at an angle to join with horizontal and vertical truss members. In other cases, part of the top and/or bottom flange is removed to make a better connection between two perpendicular beams. Piping usually has a flange at the end where two sections are joined. A reaction chamber at a process plant might have a hemispherical end cap.

The length of a structural element is specified by the coordinates at both endpoints and working point offsets. A beam connected to the side of a column can share a common endpoint coordinate where they are attached, but in reality the end of the beam is offset by half the depth of the column. The offset distance is the working point offset which effectively shortens the length of a structural element.

The location of a structural element's endpoint relative to its cross section is known as the cardinal point. Consider a column placed in the corner of a room. It is easier to specify the endpoints of the column by the location of the corner of the room, rather than the location of the center of the column relative to the corner. Frequently, compass directions are used to designate cardinal points.

2.2 Beam Prototype

A Beam prototype was written to generate VRML structural elements with the characteristics described above. The main requirements of the prototype are to provide a simple, neutral user input format for specifying a beam and to provide feedback to the user in the VRML browser about the characteristics of the beam. The simplest user input for a beam is the coordinates of its endpoints and an AISC member type designation. This simple neutral format will make it easier to generate input from a variety of external sources including CAD programs and databases. Feedback to the user within the VRML browser is generated by moving the mouse over a beam which highlights it and pops up a text display with information about the beam. This text popup is an important link to providing more data about a beam than simply a visual display of its geometry.

2.2.1 Implementation

The Beam prototype consists of a Shape node for the geometry of the structural element and a Script node to process all of the input parameters. The Script node is written in JavaScript and was developed to work properly with several popular VRML browsers.

Several Shape geometry nodes were considered including IndexedFaceSet, Box, Cylinder, and Extrusion. An IndexedFaceSet is the most general geometry node available; however, determining the coordinates of the faces would not be trivial for the general case of an I-beam with arbitrary position, orientation, and end cuts. An I-beam could also be generated from three Boxes and a pipe from a Cylinder. However, using multiple Box and Cylinder geometries would require including multiple Shape nodes in the prototype which is less desirable than using only one Shape node. The best alternative for the Shape geometry node is an Extrusion. It provides most of the necessary inputs for simply describing an arbitrary structural element and can be easily incorporated into the Beam prototype

The two main input parameters for the Extrusion node are the spine and crossSection. The spine is a list of coordinates along which the cross section is swept to create an extrusion. For a straight beam the list of coordinates are its two endpoints. However, for a beam with arbitrary endpoints, the orientation of the cross section depends on the endpoints. Figure 1 shows three rectangular Extrusions sharing a common coordinate on the spine. No orientation has been specified in the Extrusion node, yet each has a completely different orientation. This is not acceptable for modeling beams. Typically, the web of an I-beam and other similar cross sections has a vertical orientation.

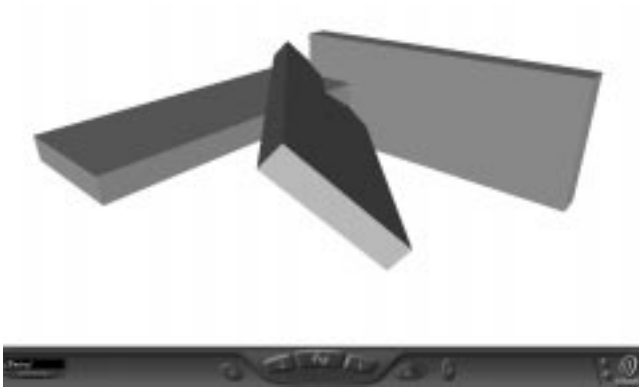

Figure 1. Extrusions with different endpoints.

To keep the web of a beam vertical, a value for the orientation field could be computed based on the endpoints, however, this gets confusing for arbitrary endpoints. A more elegant solution and one that works for arbitrary endpoints was developed. The spine of the Extrusion, as computed in the Script node, always starts at the origin and lies along the positive Y axis. The length of the spine is computed from the distance between the endpoints. The Extrusion is nested in two Transform nodes. The inner and outer Transforms are a rotation and a translation, respectively, that properly orients and positions the Extrusion based on the coordinates of the endpoints. The single rotation Transform is computed from rotations about the X and Z axes. The two rotations are combined with quaternions [3].

A lookup table in the Beam prototype contains the relevant cross section dimensions (i.e. flange thickness and width and web thickness and depth for an I-beam) for over 1000 standard AISC structural members. Given a standard structural member designation, the dimensions are retrieved and the crossSection field of the Extrusion node is generated.

2.2.2 End Conditions

An angle cut at the end of an I-beam can be modeled using the orientation field of the Extrusion node. The orientation field specifies how the cross section is rotated at each coordinate of the spine. The angle of the cut is the rotation of the cross section. However, the aspect ratio of a rotated cross section remains the same as an unrotated cross section. Therefore a beam with only one angled end will be tapered as shown in Figure 2. An Extrusion without using the orientation field is shown for comparison. To prevent a beam from being tapered, the scale field of the Extrusion node is used to stretch the cross section appropriately depending on the angle of the cut.

End conditions such as pipe flanges and hemispherical ends on cylinders are modeled by adding points along the spine of the Extrusion near either end. At those points, the cross section is scaled, using the scale field, to create the appropriate end condition geometry. One end condition that cannot be modeled by modifying the spine, orientation, or scale of the Extrusion is a notch cut from the part of the flange of an I-beam. A notch in a beam could be modeled if the beam was generated from multiple Box nodes. Notches are frequently required when connecting two beams that are perpendicular to each other.

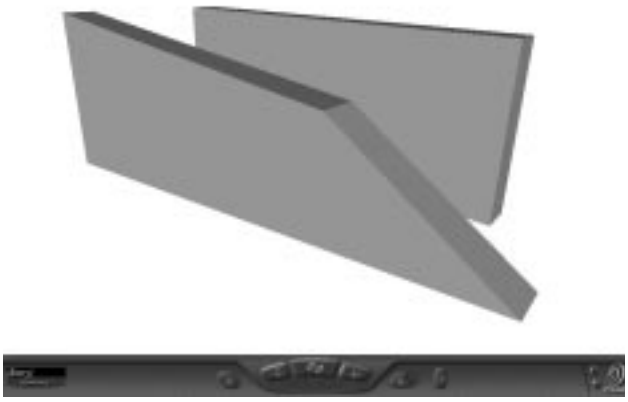

Figure 2. Extrusion with specified orientation.

2.2.3 Text Popup

The text popup is implemented in the Beam prototype with a Text node for each beam that is in a heads-up-display. The text popup construct is grouped within a Switch and a TouchSensor associated with the Extrusion. The TouchSensor output is sent to the Script which controls turning on and off the Text with the Switch and changes the color of the Extrusion to highlight it. The basic information contained in the text string are the cross section designation, beam length and weight, position, and a user-defined string. The user-defined string can be used to identify which larger assembly a beam is part of.

2.3 Beam Examples

Table 1 describes all of the user inputs of the interface definition of the Beam prototype. Not all of the parameters have been previously described. All of the parameters have default values.

An example of the VRML input for a group of beams generated with the Beam prototype is shown below. The beam parameters are defined in Table 1.

```
Group {
 children [
  # cylinder
  Beam {typ "CY"  p1 0 0 0  p2 0 0 120}
  # I-beam
  Beam {typ "W24X207"  p1 0 0 0  p2 0 120 0}
  # round pipe with flanges
  Beam {typ "P6STD"  p1 0 0 0  p2 0 0 200
        fl1 .05 .05  fl2 .05 .05}
  # square tube with angle cuts
  Beam {typ "ST3X8.625"  p1 0 0 0  p2 150 0 0
        angx1 30 angx2 45}
  # 90 degree cylindrical elbow
  Beam {typ "CY" p1 10 0 0 p2 0 10 0 p3 0 0 0}
 ]
}
```

Table 1: Beam prototype input parameters

NAME	DESCRIPTION
typ	extrusion type, one of: CYLINDER, BOX, IBEAM, TBEAM, CBEAM, LBEAM, PIPE, GENERAL (only the first 2 characters are needed) or any valid AISC section designation, i.e. W36X300
txt	additional text for popup
cp	cross section cardinal point, one of: N, S, E, W, NE, SE, NW, SW
p1, p2	coordinates of endpoints
p3	coordinates of center of radius for an elbow
bf, d, tf, tw	non-standard web and flange dimensions
fl1, fl2	pipe flange dimensions
angx1, angx2, angy1, angy2	angle of end cut about x or y axis
hcut1, hcut2, vcut1, vcut2	offset distance for horizontal and vertical end cuts
d1, d2	cylinder or pipe diameters
wpo1, wpo2	working point offsets
xsrot	clockwise rotation of cross section
cAngle	crease angle for rounded cross sections
beginCap, endCap	generate beginning or end caps
hm1, hm2	generate hemispherical ends
solid	make extrusion solid
ts	touch sensor for text popup enabled
color, specular	colors
xsection	user-defined cross section for ‘‘typ’’ GENERAL
xspine	user-defined spine

Plate 1 shows the VRML model of an emissions control system (ECS) that will be built for the fire research facility at NIST. Details of part of the structure are shown in Figure 3. The ECS was modeled almost entirely using the Beam prototype, even the buildings and foundations. The VRML file was generated by hand from 2D CAD drawings, although in the future, the model will be generated from a construction project database. The VRML model will serve as a web-based 3D view of the construction project and as a testbed for exploring the presentation of many types of information relevant to the construction process.

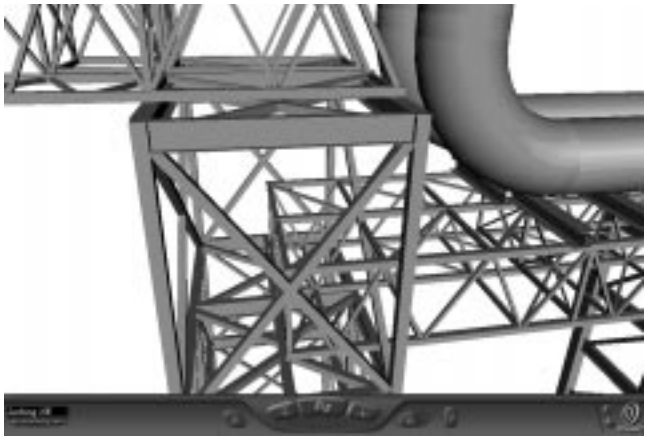

Figure 3. Details of ECS steel structure.

3 CONSTRUCTION EQUIPMENT AND MATERIALS

Construction equipment, such as excavators, dumptrucks, and cranes, are articulated machines with a hierarchy of parts connected through joints that can rotate or translate. An excavator consists of a bucket connected to a dipper arm, that is connected to a boom, that is connected to the cab, that is connected to the base. The tracks on the base allow the excavator to move. The connections between various parts allow for a rotation about one axis. The rotation at those connections are controlled by hydraulic cylinders or motors.

One of the main purposes of the construction equipment is to move materials at a construction site. The materials are either structured, such as steel beams and machinery, or unstructured, such as dirt, gravel, or concrete. The equipment operates in a dynamic environment where equipment and materials are being moved, attached together, or taken apart, and the terrain itself can change shape. For example, an excavator fills its bucket with dirt from a trench and deposits the dirt in a dumptruck which then drives away and dumps the dirt in another location. For another example, a crane lifts a load of steel beams from the back of a flatbed truck and places them on the construction site. Then the crane lifts one beam into place where it is attached to a steel structure.

Modeling the behavior, in VRML, of construction equipment and materials, described above, presents many challenges. Those include developing useful 3D geometric models of construction equipment, keeping track of the dynamic state of the construction site to avoid collisions between equipment and materials, moving materials between equipment and/or the environment, and controlling the position and orientation of equipment and materials from external data sources or within a VRML browser.

3.1 Geometry Models

Three-dimensional models of construction equipment are available for purchase from several companies and freely from some web sites. The models usually come in several different file formats (3DS, DXF, OBJ, etc.). Rarely, though, do they come in VRML format. Only a small number of models available are designed for real-time interactivity. The models almost always have too many polygons (tens of thousands) and a lot of unnecessary detail. For example, the 3D model of a dumptruck, freely available on the web, has over 50 % of all of its polygons for the entire model in just the tires and wheels. Unnecessary details such as the rear view mirrors and gas tank straps are also modeled. This leads to very large VRML files and slow graphics processing. Translating from the file formats listed above to VRML is not straightforward and can lead to poorly formatted and inefficient VRML files.

Existing models can be optimized for VRML by removing extraneous detail, replacing high polygon count parts with simpler VRML primitives (Cylinders, Boxes, or Extrusions), and using VRML optimization software [4] [6] to reduce the polygon count. Being able to articulate a model requires its part hierachy and the location of points about which they rotate. For existing models, this type information has to be added by hand.

3.2 Excavator

Many of the challenges described above can be shown in several VRML models of construction equipment and material that have been developed in this research. The following is part of the VRML file for the excavator shown in Figure 4. Each part is nested in a Transform and grouped with the part it is attached to. The center field of each Transform defines the location of the joint about which a part rotates. The individual parts of the excavator geometry were generated in a CAD program, translated to VRML, and optimized.

```
DEF EXCTRACKS Transform {
  children      [
    ... track geometry
    DEF EXCCAB Transform {
      children  [
        ... cab geometry
        DEF EXCBOOM Transform {
          children      [
            ... boom geometry
            DEF EXCDIPPER Transform {
              children  [
                ... dipper geometry
                DEF EXCBUCKET Transform {
                  children      [
                    ... bucket geometry
                  ]
                  center ...
                }
              ]
              center ...
            }
          ]
          center ...
        }
      ]
      center ...
    }
  ]
}
```

The articulation of the excavator is controlled by the widgets (levers, sliders, arrows) in the Excavator window, seen in the lower right corner of the browser window. The user-controlled widgets allow for most aspects of excavator operation. The widgets are VRML prototypes that pass a value to a Script node which in turn routes rotations and translations to various Transform nodes related to the excavator. Because the widgets are controlled by mouse movements, only one widget can be manipulated at a time unlike a real excavator where most of the controls can be operated simultaneously. For real-time monitoring of a construction site, the position and orientation of equipment can be determined from field sensor data and accessed through the External Authoring Interface (EAI). The IEEE 1278 Standard for Distributed Interactive Simulation [8] can be used to communicate field sensor data from construction equipment, including state information such as position, orientation, velocities and accelerations.

3.3 Terrain

The terrain in the excavator model is generated with an ElevationGrid which specifies a height field over a uniform grid. Using an ElevationGrid makes it easy to track the excavator on the terrain grid. As the excavator drives over the terrain, a Script node computes the appropriate orientation depending on the elevation of the grid at the center and four corners of the excavator tracks. As shown in Figure 4, the excavator is properly oriented on a slope and is digging a trench. To visually represent the digging process, the location of the bucket relative to the terrain and relative to the excavator needs to be known. The position of the bucket depends on all of the rotations of the excavator joints, the excavator geometry, and the overall position and orientation of the excavator. Given these inputs, a Script node computes the position of the bucket relative to the terrain and decides if the bucket has penetrated the terrain. If the bucket has penetrated the terrain, then the ElevationGrid can be modified to give the appearance that a trench has been dug. The Script also does a simple simulation of dumping dirt from the bucket as shown by the dirt pile just beyond the trench. The appearance of digging and dumping dirt by the excavator is only a visual simulation. It is not a physics-based simulation that takes into consideration such factors as the volume of the bucket and the physical properties of the terrain.

The simulation of digging and driving over the terrain works well in this example because the size of an individual grid in the ElevationGrid is smaller than the footprint of the excavator. A finer grid will yield a better visual simulation of digging and driving. However, if the grid size is larger than the footprint of the excavator or bucket, then the driving and digging simulation will not be as accurate. Frequently, terrain is modeled as an IndexedFaceSet of nonuniform triangular facets and the methods described above for digging and driving cannot be used. Methods have been developed for following terrain constructed from arbitrary polygons [2]; however, simulating digging in such an environment would be very complex.

3.4 Construction Site

The VRML construction scene in Plate 2 consists of the excavator and dumptruck, previously mentioned, and a tower crane and concrete truck. The concrete truck was modeled by removing the bed of the dumptruck and replacing it with appropriate geometry. In this way, many different types of trucks can be constructed by reusing basic geometric elements. The individual members of the tower crane were not modeled explicitly. Only the four sides of the tower were modeled and a texture image of the truss members was mapped to the sides. This results in a model with very few polygons and a high level of detail. The hook of the tower crane is a more complex IndexedFaceSet.

The excavator, dumptruck, and tower crane in the construction scene have user-controlled widgets similar to those shown in Figure 4. The potential interaction between these objects illustrates several issues for user-controlled virtual environments. Collision detection between the objects is necessary for them to have realistic interactions. VRML does not implement collision detection between objects, only between the viewer and the virtual world. The V-COLLIDE architecture [7] has been proposed to handle collision detection between objects but has not been implemented in any VRML browser. The shape of the objects that might be involved in collisions is also dynamic. A bounding box around the excavator changes shape as the position of the bucket changes. Materials can also change their parent-child relationship in the scene graph. The dirt that was once a child of the dumptruck is now a child of the ground. Moving materials between objects can be implemented with the addChildren and removeChildren

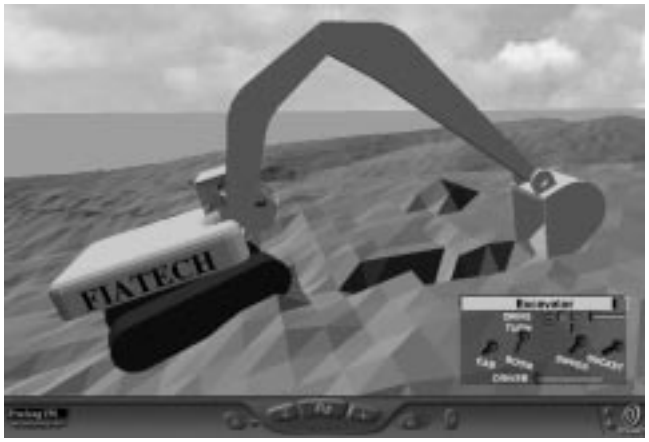

Figure 4. Excavator model on uneven terrain while digging.

methods of a Group node. Keeping track of object collisions and parent-child relationships is not necessary when the position and orientation of objects is controlled by field sensor data. Presumably, the sensor data is accurate so that no objects will appear to have collided.

4 CONCLUSIONS AND FUTURE WORK

The research described in this paper has shown that VRML can successfully be used in construction industry applications such as the modeling of steel structures and construction equipment. Several challenges have been overcome through the use of a Beam prototype and novel solutions to user-controlled construction equipment operation.

Future work will improve the implementation of the Beam prototype; create prototypes for other useful objects; and create a VRML-based user interface for the construction project of the emissions control system at NIST. The interface will access a database of as-designed and near real-time as-built construction project information. The database will contain information about the position and orientation of structural steel members and other materials on the construction site. Materials will be tracked as they arrive on site through their final assembly. The progress of work at the construction site will be reflected in a continually changing VRML model. From the VRML model, the database will be queried to access other non-geometric information about the construction project.

Object-oriented CAD systems are emerging into the construction industry design practice. These systems treat geometry as just another attribute, in this case the shape, of objects. In these systems, models are composed directly out of construction objects such as beams, columns, slabs, doors, etc., rather than geometry objects such as extrusions, boxes, or faces to which object attributes are attached.

At the same time, new data exchange protocols are emerging that enable the extraction of complete models in the form of object instances. Three protocols based on product data modeling are being explored in this research. ISO 10303, the Standard for the Exchange of Product Data (STEP) [10], includes several application protocols relevant to the construction industry, notably in the areas of ships and process plants. The CIMSteel Integration Standards/2 [5], recently adopted by the AISC, applies STEP technologies specifically to the description of structural steelwork from design and analysis through detailing and fabrication. Finally, the Industry Foundation Classes (IFC) [9], being developed by the International Alliance for Interoperability, employ STEP technolo-

gies in the definition of objects, and their interfaces, suitable for use in architecture, engineering, and construction systems. Post-processing information from any of these protocols into an intelligent VRML model will be a relatively straightforward task, making VRML a natural choice for externally viewing construction models created in the new systems.

References

[1] AISC, American Institute of Steel Construction, Load and Resistance Factor Design Manual of Steel Construction, 2ed., 1998.

[2] J.W. Barrus, R.C. Waters, "QOTA: A Fast, Multi-Purpose Algorithm For Terrain Following in Virtual Environments", in *VRML97 Second Symposium on the Virtual Reality Modeling Language*, pp. 59-64.

[3] G. Bell, A program that condenses multiple rotations into one rotation field, URL: *http://hiwaay.net/~crispen/src/rotations.zip*

[4] Chisel, Trapezium Development LLC, URL: *http://www.trapezium.com/Chiselpage.htm*

[5] CIMSteel Integration Standards, Release 2, URL: *http://www.cis2.org*, The Steel Construction Institute, Ascot, Berkshire, UK, URL: *http://www.steel-sci.org*.

[6] Flamingo Optimizer, Novafex Software Limited, URL: *http://www.novafex.com/prod_fo.html*

[7] T.C. Hudson, M.C. Lin, J. Cohen, S. Gottschalk, D. Manocha, "V-COLLIDE: Accelerated Collision Detection for VRML", in *VRML97 Second Symposium on the Virtual Reality Modeling Language*, pp. 117-123.

[8] IEEE 1278.1 Standard for Distributed Interactive Simulation, IEEE.

[9] Industry Foundation Classes Release 2, The International Alliance for Interoperability, 1999, URL: *http://iaiweb.lbl.gov*

[10] ISO 10303, Industrial Automation systems and integration - Product data representation and exchange. ISO 10303-227, Plant Spatial Configuration, ISO Central Secretariat, Geneva, Switzerland, URL: *http://www.iso.ch*, and American National Standards Institute, New York, URL: *http://www.ansi.org*, STEP development information, URL: *http://www.nist.gov/sc4*.

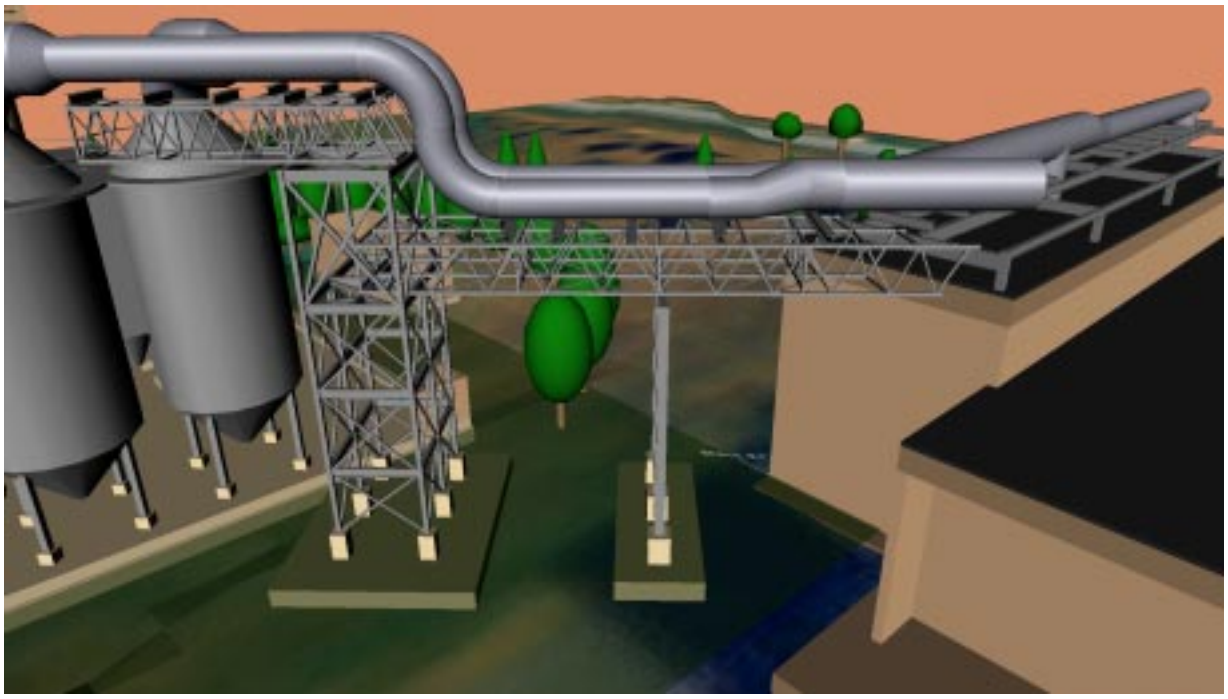

Plate 1: Steel structure modeled with the Beam prototype.

Plate 2: Construction site scene.